BLOOD PRESSURE MEASURING DEVICE EMBEDDED WITH SMS CAPABILITIES

By

SITI HAJAR BINTI ANUAR

FINAL REPORT

Submitted to the Electrical & Electronics Engineering Programme
in Partial Fulfillment of the Requirements
for the Degree
Bachelor of Engineering (Hons)
(Electrical & Electronics Engineering)

DECEMBER 2009

Universiti Teknologi Petronas
Bandar Seri Iskandar
31750 Tronoh
Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

BLOOD PRESSURE MEASURING DEVICE EMBEDDED WITH SMS CAPABILITIES

by

Siti Hajar binti Anuar

A project dissertation submitted to the
Electrical & Electronics Engineering Programme
Universiti Teknologi PETRONAS
in partial fulfilment of the requirement for the
Bachelor of Engineering (Hons)
(Electrical & Electronics Engineering)

Approved:

____

Assoc. Prof. Dr. Irraivan Elamvazuthi
Project Supervisor

UNIVERSITI TEKNOLOGI PETRONAS
TRONOH, PERAK

December 2009

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

SITI HAJAR BINTI ANUAR

ABSTRACT

For a high blood pressure patient, checking their blood pressure at home is an important part in managing high blood pressure (hypertension). It is recommended that anyone with high blood pressure to monitor their blood pressure at home. This home monitoring can help them keep tabs on their blood pressure in a familiar setting, make certain their medication is working, and alert the patient and the doctors to potential health complications. Doctors can use the measurements to see how well the medicine is working to control the high blood pressure. The current blood pressure measuring device in the market has the functions to display the blood pressure, heart beat rate and save the obtained data in the device memory. But so far, there is no development yet to this device where these obtained data can be sent to other people for records. Thus, this project presents the development of a blood pressure measuring device that is embedded with SMS capabilities.

ACKNOWLEDGEMENTS

First and foremost, I would like to express my gratitude to my Final Year Project supervisor, Assoc. Prof. Dr. Irraivan Elamvazuthi and also fellow final year students for their invaluable help, advices, suggestions, and encouragement. Their untiring effort has been a continuous inspiration to me.

My utmost and heartfelt thanks are to all the people who have spent time to consult with me regarding the concept and methodologies of this project in order for me to smoothly understand and implement this project. I would like to thank the FYP technician, especially Mrs. Siti Hawa for helping me in preparing the entire necessary document in order to make sure that our final year project is running smoothly.

Most importantly, I would like to thank my parents who love me, support me and teach me to be a responsible and wise person. Their sacrifice and support throughout my life have helped me a lot in my education.

Last but not least, I would like to extend my appreciation to my classmates in University Technology PETRONAS (UTP) who have support me morally throughout this project. I would also like to express my faithful thanks to all whose their name had not mention, but have either directly or indirectly assists me in this project. All your helps, kindness, valuable guidance and advice will never be forgotten.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

GSM	Global System for Mobile Communication
LCD	Liquid Crystal Display
MCU	Microcontroller Unit
SMS	Short Messaging Service
PCB	Printed Circuit Board
PHP	Hypertext Processor
MYSQL	My Structured Query Language
VB	Visual Basic

CHAPTER 1
INTRODUCTION

1.1 Background of Study

This blood pressure measuring device consists of three main parts: external hardware (such as cuff, motor, valve, and LCD), analog circuit, microcontroller and a GSM modem. The analog circuit converts the pressure value inside the cuff into readable and usable analog waveforms. The MCU (Microcontroller Unit) that acts as the microcontroller will sample the waveforms and performs the A/D conversion so that further calculation can be made. The MCU also controls the operation of the devices such as the button and LCD display.

Usually when the doctor measures the patient's blood pressure, he will pump the air into the cuff and use the stethoscope to listen to the sounds of the blood in the artery of the patient's arm. At the start, the air is pumped to the above systolic value. At this point, the doctor will hear nothing through the stethoscope. After the pressure is released gradually, at some point, the doctor will begin to hear the sound of the heart beats. At this point, the pressure in the cuff corresponds to the systolic pressure.

After the pressure decreases further, the doctor will continue hearing the sound (with different characteristics). And at some point, the sounds will begin to disappear. This is where the pressure in the cuff corresponds to the diastolic pressure. This is just the example of the method that doctors normally use but in this project hearing this sound would not be used since it will be replaced by a sensor that is a built-in in the designed device [1].

This systolic and diastolic pressure measurement will take place and the result of those measurements will be displayed on the LCD. Then as the development of this device, these results will be automatically recorded and sent to other people via SMS (Short Message Service) using the GSM modem [2].

1.2 Problem Statement

The current blood pressure monitoring device so far only has the functions to display the blood pressure and heart beat rate and also they can save the data obtained in the memory inside the device. But so far in the market, there is no development yet to this device where this obtained test data can be sent to other people for records for instance doctors or the loved ones who cares for the blood pressure patients.

So, this project is designed since it is hard to keep track of the obtained daily results of the blood pressure test done especially for those who live apart. The development is to have an extra function for the device that is the device is able to send the daily test results to other people via the Short Messaging Service (SMS).

This is not just beneficial for the patients but also the doctors where they can monitor their patient’s current condition daily even though the patient is living at home and not under the doctor’s supervision in the hospital.

1.3 Objectives

The objective of this project is to develop a blood pressure monitoring device so that the test result obtained can be sent to other people via the Short Messaging Service (SMS). This is beneficial especially for doctors to keep track of their patient's daily health.

1.4 Scope of Study

The scope of the project:

- Understand the "oscillometric" method, the process and how it works.
- Understand the existing pressure measurements methods and how that can be developed and implemented in the project.
- Understand the GSM modem and how they work so that SMS can be sent to mobile phones by using this modem.

CHAPTER 2
LITERATURE REVIEW

2.1 Blood Pressure Measuring Device

The current market blood pressure monitoring device manages to measure the blood pressure and heart beat rate. Most clinics and hospitals are using the conventional way of measuring the blood pressure which is by using stethoscope.

Usually when doctors measure patient's blood pressure, they will pump the air into the cuff and use the stethoscope to listen to the sounds of the blood in the artery of the patient's arm. At the start, the air is pumped to the above systolic value. At this point, the doctor will hear nothing through the stethoscope. After the pressure is released gradually, at some point, the doctor will begin to hear the sound of the heart beats. At this point, the pressure in the cuff corresponds to the systolic pressure [3].

After the pressure decreases further, the doctor will continue hearing the sound (with different characteristics). At some point, the sound will begin to disappear. This is where the pressure in the cuff corresponds to the diastolic pressure. The systolic and diastolic pressure can be seen in figure 1.

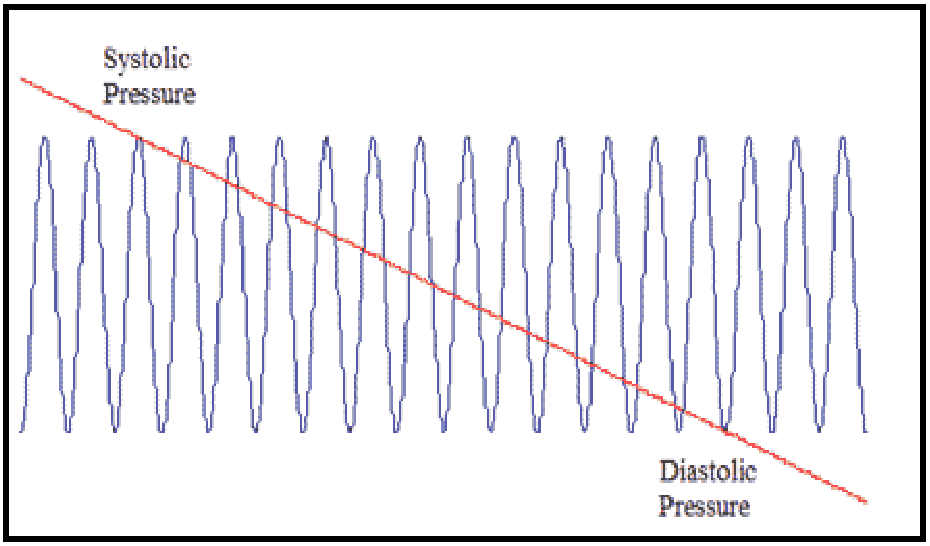


Figure 1 : The Systolic and Diastolic Pressure Measurement [4]

In this project, hearing of the sound would not be used since it is replaced by a pressure transducer that is built-in in the designed device and stethoscope is not used.

2.1.1 Hardware Diagram for Blood Pressure Measuring Device

Figure 2 shows how the device is operated. The user will use buttons to control the operations of the whole system. The MCU is the main component that controls all the operations such as motor and valve control, A/D conversion, and calculation, until the measurement is completed. The results then are is displayed at the LCD screen for the user to see.

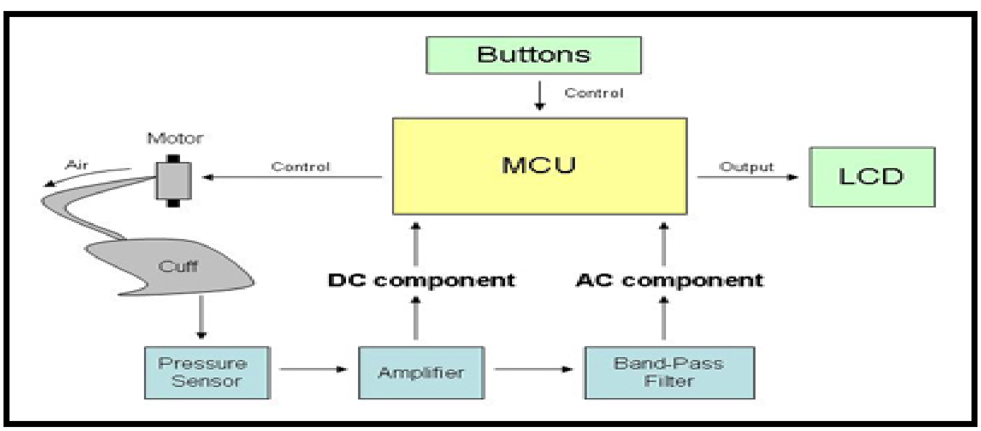


Figure 2 : Hardware Diagram for Blood Pressure Monitoring Device [1]

2.1.2 Features to Consider

It is known that the features on home blood pressure monitors can vary widely, from simple manual models to top-of-the-line fully automated devices that allow patients to send data to the respective in charged doctors through the phones. Before carrying out the prototype, there are features that are needed to be considered.

2.1.2.1 Cuff Size

The most important factor to consider when designing a blood pressure monitoring device is to have a properly fitting cuff. It has been widely produced in the market with different-sized cuffs to fit different-sized arms. This is an important feature to be considered as poorly fitting cuffs will not give accurate blood pressure measurements.

2.1.2.2 Display

The display that shows the blood pressure measurement should be clear and easy to read so that it can prevent from mistakes of obtained results. In this project the display would be the Liquid Crystal Display (LCD) monitor. This LCD is chosen since it can depend on battery for power and uses very small amounts of electric power.

2.2 SMS Technology Used

2.2.1 Hypertext Processor (PHP)

PHP or Hypertext Processor will be the language for developing the system. PHP is being used wisely to develop commercial software by many developers. Below is the comparison between PHP [5] and VB.net [6]. Table 1 shows the comparison of VB and PHP.

Table 1 : Comparison of VB and PHP

VB.net	**PHP**
❖ VB has been built from the ground up as an object oriented language. ❖ VB has a number of security and reliability features built right into its core design ❖ Steadily improved in runtime performance, better than C and C#.	❖ Low maintenance and development course. ❖ High performance and reliability. ❖ Able to embed itself into HTML code. ❖ Compatible with server like Apache and IIS.

Both languages are strong and have their own capabilities. Besides, many improvements in their reliability and capability performance have made PHP and VB become software developer choices.

But in this project, PHP is selected as it has low maintenance and development cost. It is open source software and can be downloaded from the internet. It can support many databases including MYSQL. MYSQL will be the database for this system. The other reason of choosing PHP is that the integration with Ozeki (SMS server) is relatively easier using PHP compared to VB.

2.2.2 Windows

Windows is the platform for running this system. Windows has become the most preferable platform since it was being introduced in 1985 [7]. People are reluctant to learn and change to new operating system because they are comfortable with the Windows.

Many applications run best in Internet Explorer (IE). This is because IE is more secured compared to Mozilla. It has more security features that can meet users need. Like many Microsoft features, IE has a quite rich set of security features that can be configured to suit the user's needs [8]. Firefox, by comparison, is more simplistic in its security configuration choices. Also, Windows can support most hardware and software. Most of the software or hardware in the market is compatible with Windows. Many open source applications (such as Open office, the Mozilla Firefox browser etc.) are also compatible with Windows [9].

2.2.1 SMS

SMS will be the notification medium for this system. Another method of sending information is via email. Both medium have pros and cons. Short Messaging Service (SMS) is being used worldwide and becomes a common technology. People prefer to text rather than emails because it is an efficient technology that can be used anytime and anywhere.

Table 2 : Advantages and Disadvantages of Email and SMS [10,11]

Email	**Short Messaging Service (SMS)**
Advantages: 1) No cost incurs. 2) Can be use to send letter, files, data, reports and etceteras. 3) Can be deal at any convenient time 4) The cost is not affected by the distance or size of the message.	Advantages: 1) Can be use for quickly communication at anytime and anywhere using mobile. 2) More efficient than Email because you can check your SMS even while you outside. 3) Fast delivery compared to Email.
Disadvantages: 1) It is not necessary private because people can intercept or read your email as it pass trough networks.	Disadvantages: 1) Message transmitted does not guarantee that it will successfully deliver because some message may be delayed, blocked or lost in the middle of transmission. 2) SMS has limitation of words. Only 160 characters only per message.

SMS can be a fast information transmission medium as it can be access whenever user wants it. But SMS has words limitation. Only simple or short information can be transmitting using this text message. In this project, that would not be a problem since data sent is not long. Thus, SMS in chosen in this project. Table 2 shows the comparison of email and SMS.

2.2.1 Group System for Mobile Communication (GSM)

Table 3 : GSM vs. 3G [12,13]

GSM	**3G**
1) Widest service distribution network and choice of hand phone 2) Low cost entry hand phone 3) Lower subscriber acquisition cost. 4) International roaming 5) Easy subscription process. 6) data transfer speeds of up to 9.6 Kbit/s	1) More bandwidth, security and reliability. 2) Rich multimedia services 3) High cost for upgrading bas station and cellular infrastructure. 4) Data transfer rate up to 128 Kbits/second for fast moving devices, such as handsets in moving vehicles 5) higher power requirements (more bits with the same energy/bit) demand a larger handset, shorter talk time, and larger batteries)

Table 3 shows the comparison between GSM and 3G. Based on the comparison table between 3G and GSM, 3G is better in terms of performance. It can transfer data faster than GSM and able to provide variety of multimedia services as nowadays a high demand of the new technology is increase. In this project, GSM is chosen to be used as it is cheaper to be implemented and has a wider service distribution network. Cost for project development can be reduced by using GSM.

2.2.1.1 *Basic Idea of Application*

This GSM modem is an external modem device. A GSM SIM card is inserted into this modem, and then this modem is connected to an available serial port on a computer. A standard GSM mobile phone with the appropriate cable and software driver is used and is connected to the serial port on the computer.

When GSM is installed, the Windows modem driver from the device manufacturer is installed. To simplify configuration, the SMS Gateway (communication network) will communicate with the device via this driver. Windows diagnostics can also be used to ensure that the modem is communicating properly with the computer.

Then, the gateway will attempt to initialize the modem, and will also confirm that the modem supports the interfaces to send and receive SMS messages. Then the SMS containing the test results from the device will reach the mobile phone of other people. This is shown in figure 3.

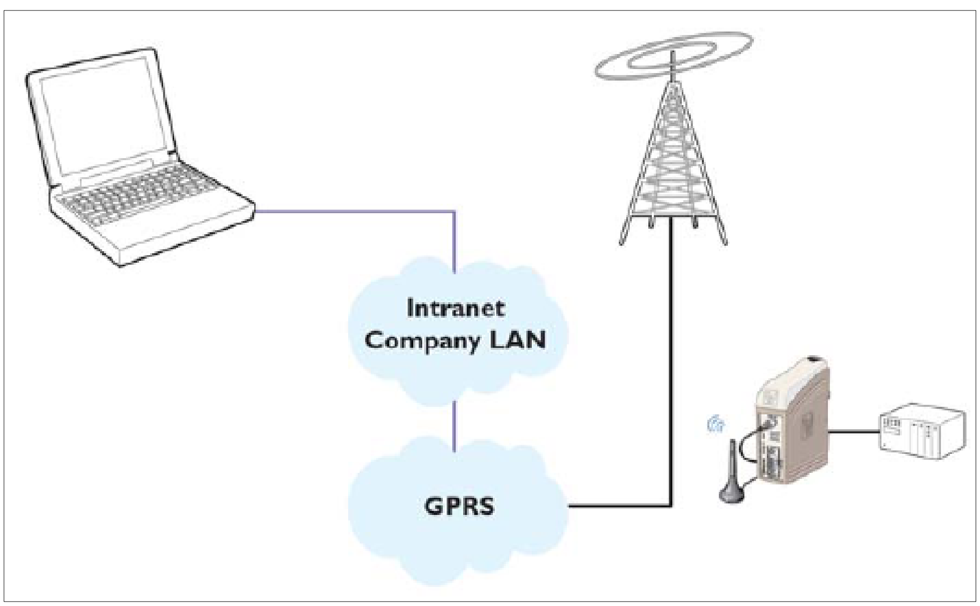


Figure 3 : The GSM Modem Application (Communication Network) [14]

2.3 Types of Blood Pressure Measurement Method

There are two types of the blood measurement method which are the auscultatory method and the oscillometric method. Table 4 shows the comparison for both methods:

Table 4 : The Comparison of the Blood Pressure Measurement Method [15,16,17]

Auscultatory Method	Oscillometric Method
• Accuracy of this method relies on the ability of the human ear to detect and distinguish sounds. • Experienced clinicians can determine the quality of each measurement. • However, inexperienced clinician may be inconsistent in their assessment of the Korotkoff sounds and be unable to distinguish them from outside noise or other interference. • Measures the systolic and diastolic pressure, and the mean is normally calculated based on the following formula MAP = (SYS + 2DIA)/3, which can be inaccurate in certain situations.	• Oscillometric devices measure the mean pressure and derive the systolic and diastolic pressures. • Almost all blood pressure monitoring devices nowadays are using this method due to its accuracy.

Oscillometric method is chosen as the blood pressure measurement method in this project due to its accuracy.

2.4 The Blood Pressure Measurement Method

The blood pressure measurement method used in this project is the ‘oscillometric’ method. The artery induced pulsations are different. When the artery is compresses, no pulsation is perceived by the device, and then when the pressure decreases in the cuff, the artery starts to emit pulsations. The pressure that has been measured on the device defines the maximal blood pressure or also known as the systolic blood pressure.

When the pressure decreases in the cuff, the oscillation will become increasingly significant, until maximum amplitude of these oscillations defines the average blood pressure. The oscillations can still be seen during the decrease of the pressure in the cuff until they disappear. The pressure then will read on the device that defines the minimal blood pressure or diastolic blood pressure.

This method of blood pressure measurement is called the oscillometric method. It is very often used in the automatic device for the measurement of the blood pressure due to its excellent reliability [15]. Figure 4 below shows the measurement method of this oscillometric method.

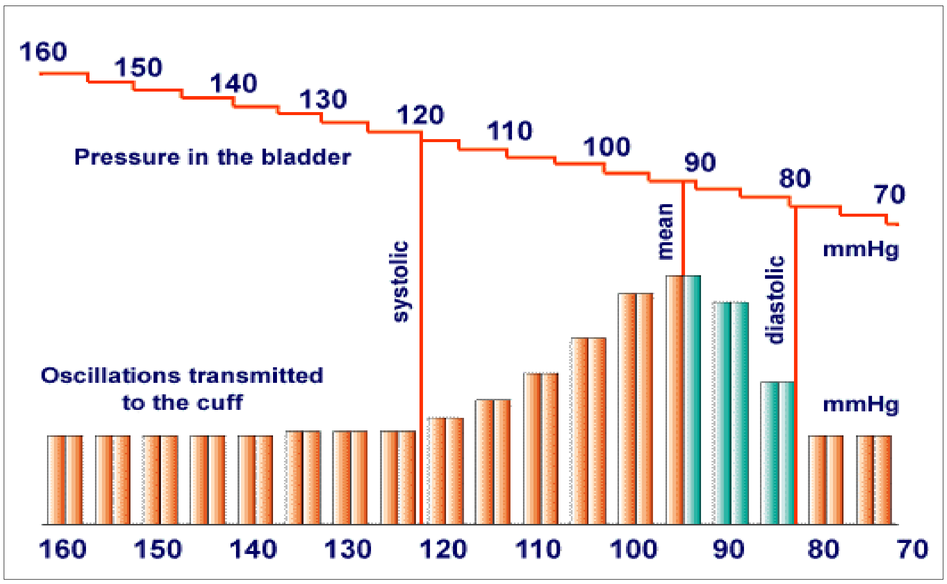


Figure 4 : Hardware Diagram for Blood Pressure Monitoring Device [15]

CHAPTER 3

METHODOLOGY

3.1 Overall Project Flow Chart

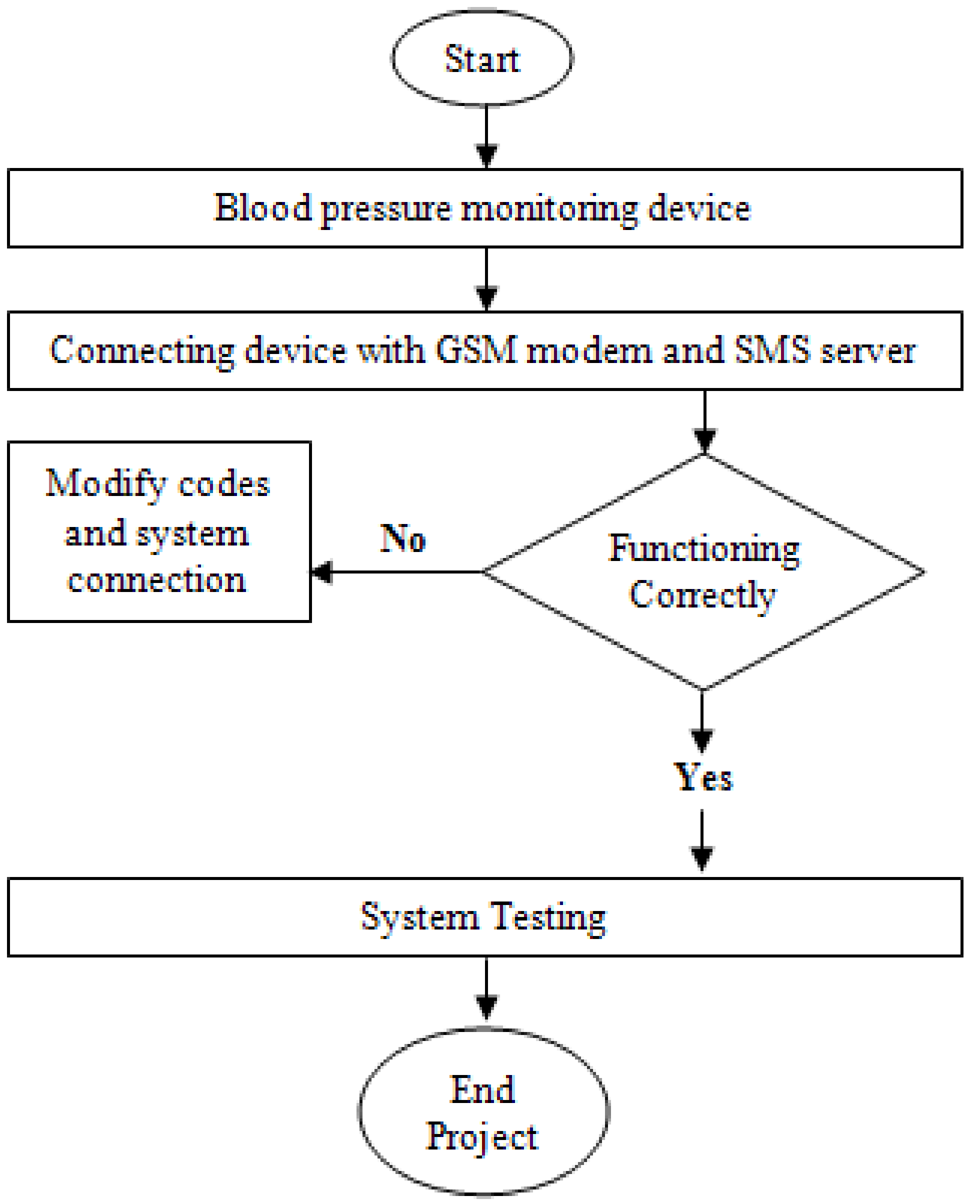


Figure 5 : The Project Flow Chart

Figure 5 shows the overall system flowchart. First, the blood pressure is connected with the GSM modem and the SMS server. If the system is functioning incorrectly, the process will be repeated and improvement is done by modifying codes and system connection. If the system is functioning correctly, then it will continue with the system testing and end up with the end user acceptance testing (test to check the end user receives the information in the SMS form).

3.2 System Configuration

Basically the overall design consists of the blood pressure monitoring device. Then, the test results obtained is sent to other people via the GSM modem in the network. Lastly, the data end receiver is the mobile phones and data is received in the form of SMS. Figure 6 shows the system configuration for this project [18, 19].

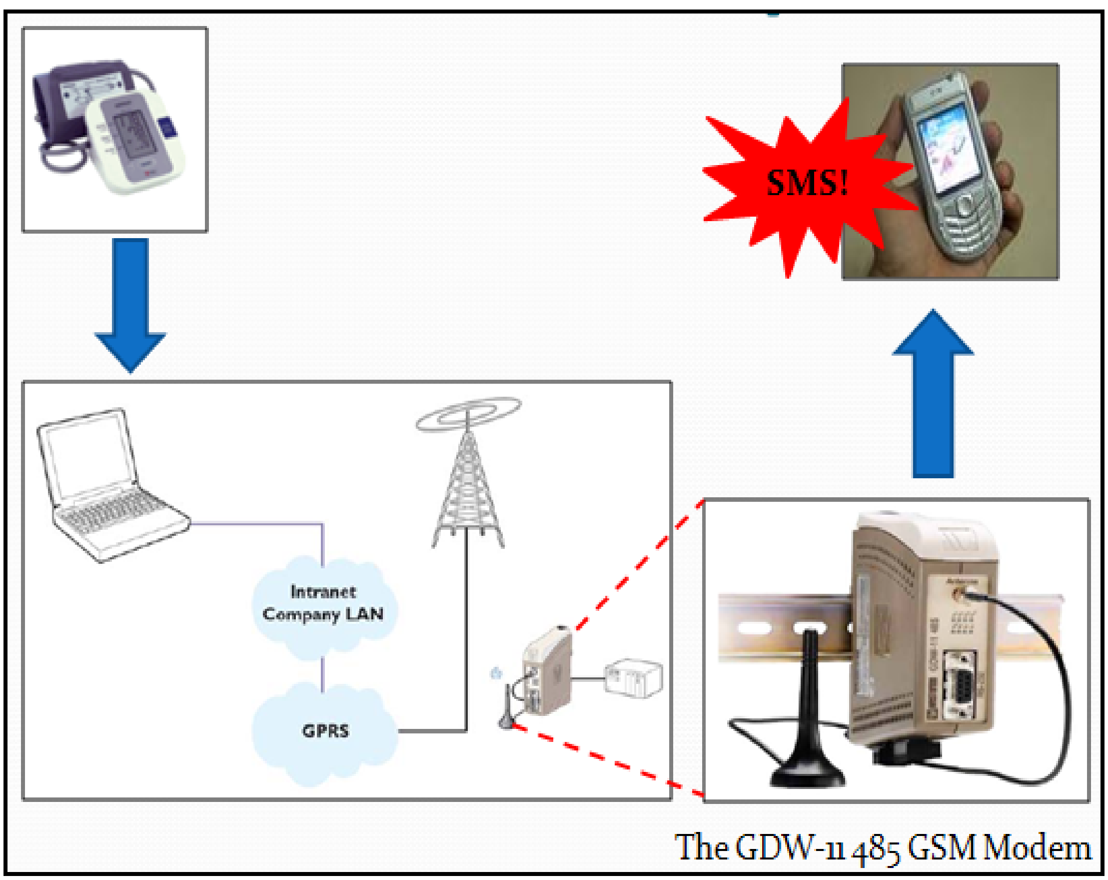


Figure 6 : System Configuration

3.3 Blood Pressure Measurement: Oscillometric Method

Figure 7 shows the oscillometric method flow of measurement. There are two types of pressure in the blood pressure measurement which are the systolic and diastolic pressures. Firstly, the device will measure the changes in the arterial counter pressure. This is imposed by the inflatable cuff which is controllably relaxed or inflated.

Then, the cuff pressure is reduced to a predetermined increment where at each pressure level fluctuation is monitored. The resulting signal will consist of the DC voltage with variation component caused by arterial blood pressure pulsation.

Then, a suitable filtering will reject the DC voltage component to provide amplification. The peak pulse amplitudes above a given base line are measured and stored. The mean arterial pressure is calculated where the systolic and diastolic pressure can be derived. This flow of process is simplified into figure 7 for better understanding.

Measure changes in the arterial counter pressure
(Imposed by the inflatable cuff which is controllably relaxed or inflated)

Cuff pressure is reduced to predetermined increment

At each pressure level fluctuations are monitored

The resulting signals typically consist of the DC voltage with variation component caused by arterial blood pressure pulsation

DC voltage component is rejected using suitable filtering & to provide amplification

Peak pulse amplitudes above a given base line are measured and stored

The lowest cuff pressure at which the oscillation has a maximum value = mean arterial pressure

Systolic & diastolic pressures can be derived as fraction of mean arterial pressure

Figure 7 : Oscillometric Method Flow of Measurement

3.4 Tools and Equipments Used and Costing

Table 5 and 6 shows the tools and equipments used in this project and the costing of each of them. There are two separate costs which are the direct and indirect costs. Indirect cost is the cost that is not spent in this project since they are available in lab and the electronic store.

Table 5 : The Tools and Equipments Used (Indirect Cost)

Tools and Equipment	Cost (RM)
1) PC	1200.00
2) Transistor – lab	6.23
3) Resistor and Capacitor - lab	44.50
Total	**1250.73**

Table 6 : The Tools and Equipments Used (Direct Cost)

Tools and Equipment	Cost (RM)
1) Digital Blood Pressure Cuff	50.00
2) PR22 (Pressure Sensor Reading Interface)	90.00
3) Programmer Cable	30.00
4) Needle-nose pliers	7.00
5) Energizer 9V Battery	19.80
6) 9V Battery Snap	0.90
7) Tweezer	4.00
8) Soldering Paste	3.00
9) Liquid Flux Remover (30 ml)	5.00
10) Bread Board	22.00
11) Soldering Iron Holder	4.00
12) LM741	2.00
13) Digital Blood Pressure Cuff	2.50
Total	**240.20**

3.5 Software and Tools Used

Here are the softwares and tools used in this project that includes:

- Hypertext Processor (PHP) – Language used to develop the system.
- Windows – Platform to run the system.
- Group System for Mobile Communication (GSM) – Used as a medium of data transfer.
- C++ Programming Language

CHAPTER 4
RESULTS AND DISCUSSION

4.1 Overall System Configuration

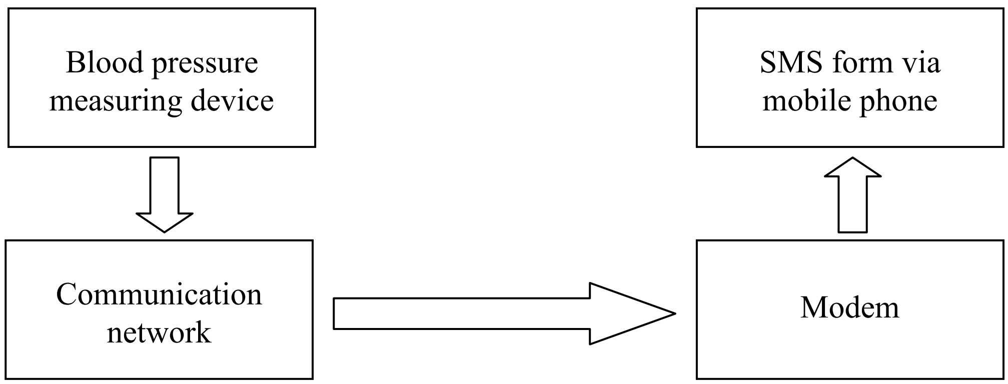


Figure 8 : The Overall System Configuration

Figure 8 shows the overall project design. The test results obtained from the blood pressure measuring device is then sent to others via the GSM modem on the network. To further explain the design, it will be divided into two parts which are:

1) The blood pressure measuring device.
2) The SMS system architecture (Communication Network).

4.2 Blood Pressure Measuring Device

Figure 9 shows the overall design of the blood pressure measuring device. There are three main parts involved which are the:

- The external hardware (Button, LCD and cuff)
- Microcontroller Unit (MCU)
- Analog circuit (Pressure sensor, amplifier, band-pass filter)

First, the user will use buttons to control the operations of the whole system. The MCU (microcontroller unit) is the main component that controls all the operations such as motor and valve control, A/D conversion and calculation, until the measurement is completed. The results then are displayed at the LCD (Liquid Crystal Display) screen for the user to see. The flow of system process is summarized in figure 9.

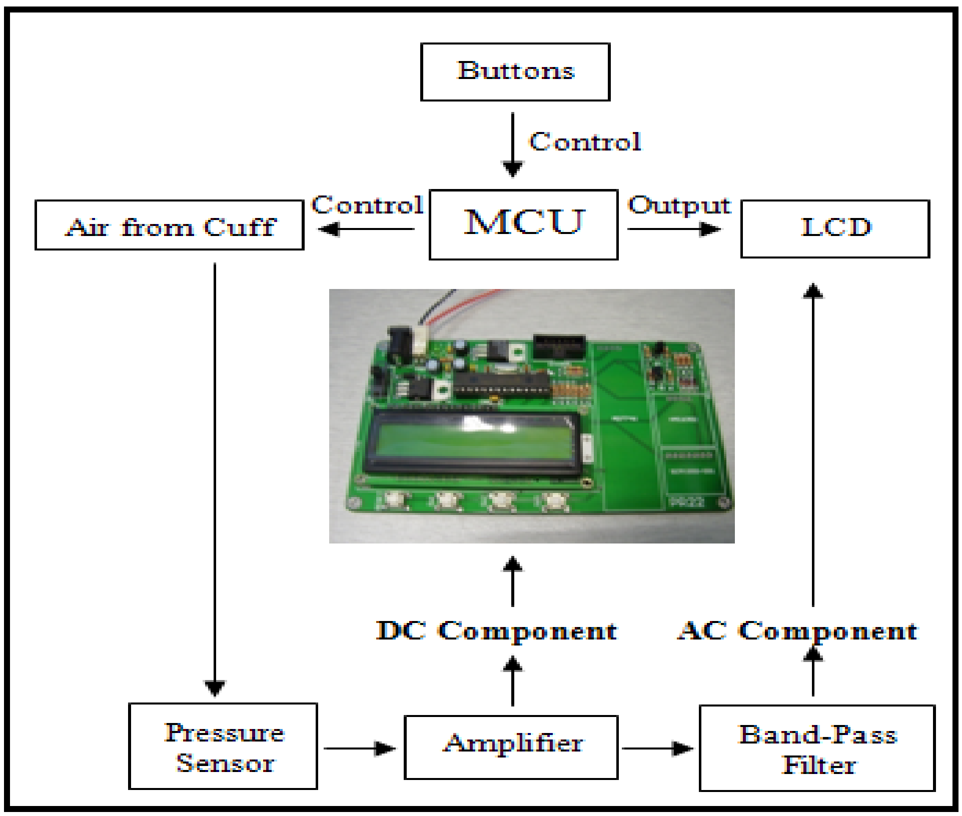


Figure 9 : Blood Pressure Measuring Device Design

4.2.1 Analog Circuit

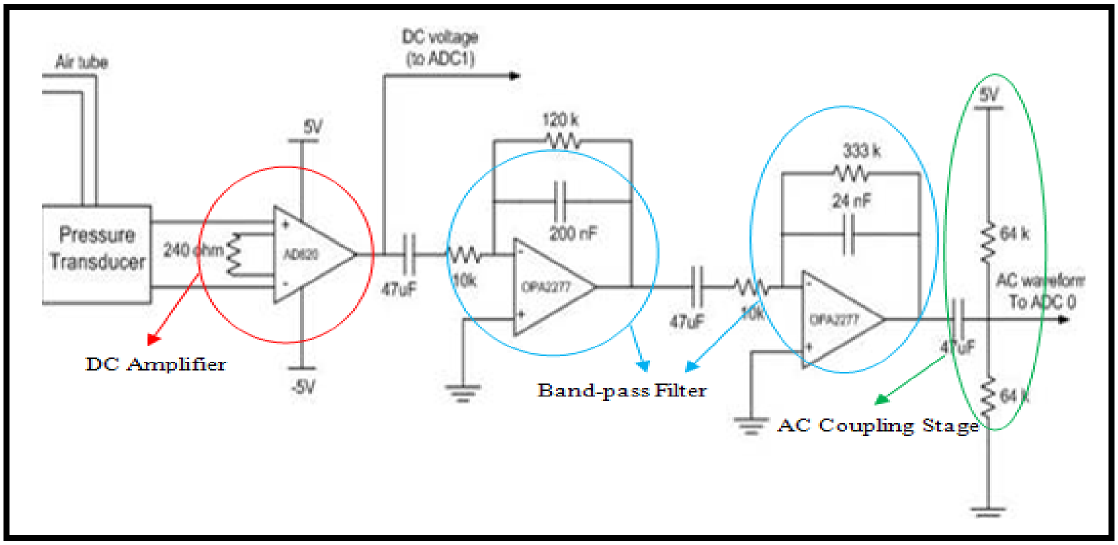


Figure 10 : Schematics of Analog Circuit Schematic

Figure 10 show the device's analog circuit which are consisted of four parts. It starts form the pressure transducer and the data obtained will be passed on according to the sequence which is into the DC amplifier and lastly at the AC coupling stage. Figure 11 shows the analog circuit constructed. This analog circuit consists of four circuits which are:

- Pressure Transducer
- DC Amplifier
- Band-pass Filter
- AC Coupling Stage

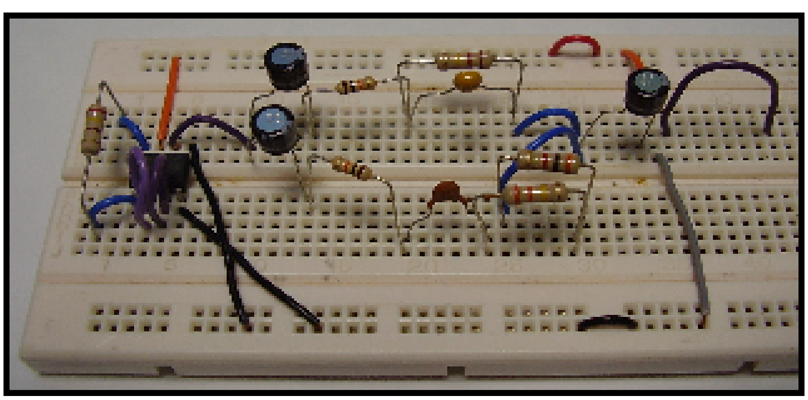


Figure 11 : The Analog Circuit Constructed

4.2.1.1 Pressure Transducer

The pressure transducer is used to sense the pressure from the arm cuff. This pressure transducer produces the output voltage proportional to the applied differential input pressure. The tube from the cuff is connected to one of the inputs and the other input is leave to open. This is done so that the output voltage will be proportional to the difference between the pressure in the cuff and the air pressure in the room. Figure 12 shows the transfer characteristics.

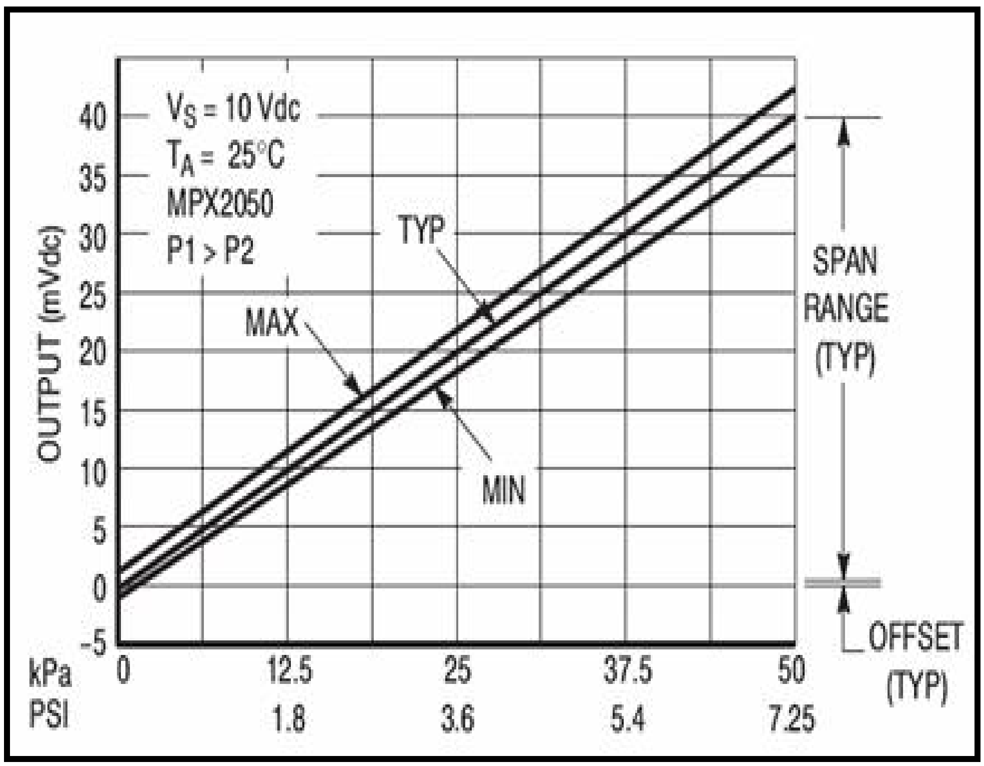


Figure 12 : Output Voltage vs. Differential Input Pressure [20]

Figure 13 shows the interface for the pressure sensor where it can read the pressure sensor and display the reading on the LCD.

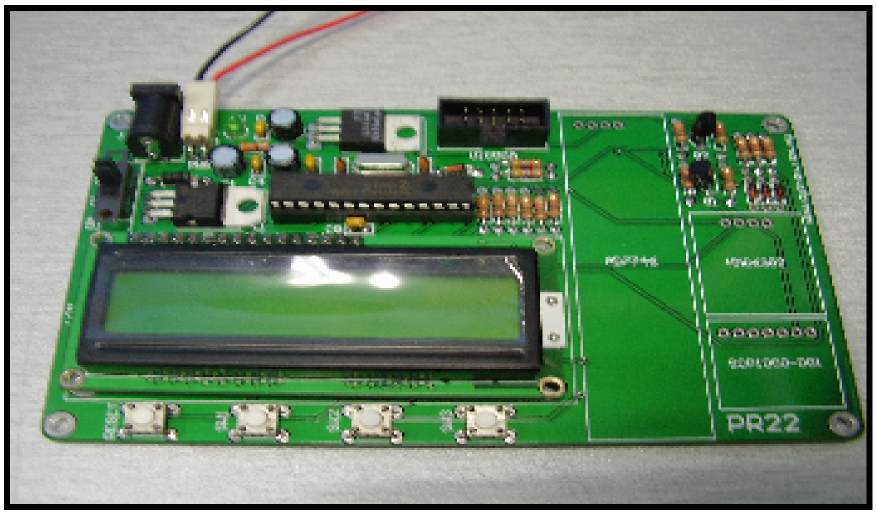


Figure 13 : Interface for Pressure Sensor

Figure 14 below shows the system overview on how this interface works. The schematics and hardware list is as attached in Appendix F. The programming in microcontroller is as attched in Appendix C.

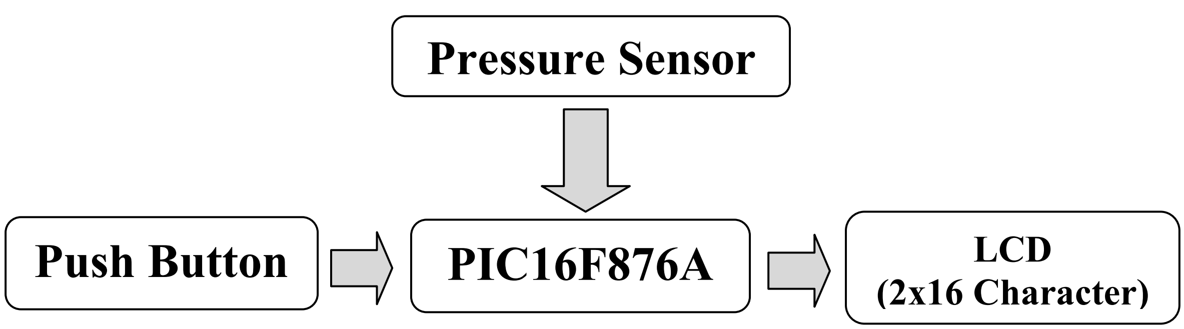


Figure 14 : System Overview of Interface

4.2.1.2 DC Amplifier

The DC amplifier is used to amplify both the DC and AC components of the output signal of pressure transducer so that further signal process by MCU can be done and data of user can be obtained. The schematic and specification of the amplifier is attached in Appendix E and G. In this project, the arm cuff is pumped only until 160 mmHg (approximately 21.33 kPa). The values of this part are:

- Output voltage = 0 to 4V
- Gain = 200

4.2.1.3 Band-pass Filter

Next, the signal will be passed on to the band-pass filter. Before going into the filter, the DC amplifier amplifies both the DC and AC component of the signal. This band-pass attenuates any signal that is out of the pass band. This AC component from the band-pass filter determines when to capture the systolic/diastolic pressures and when to determine the heart rate of the user. The values are:

- Gain = 1 - 4 Hz

4.2.1.4 AC Coupling Stage

Lastly is the AC Coupling stage where this circuit is used to couple only the AC component of the signal so that it can provide the DC bias independently. The AC output from this stage is passed on to the analog-to-digital converter in the microcontroller. The sine wave can be produced and seen in an oscilloscope in lab.

4.2.2 SMS System Architecture

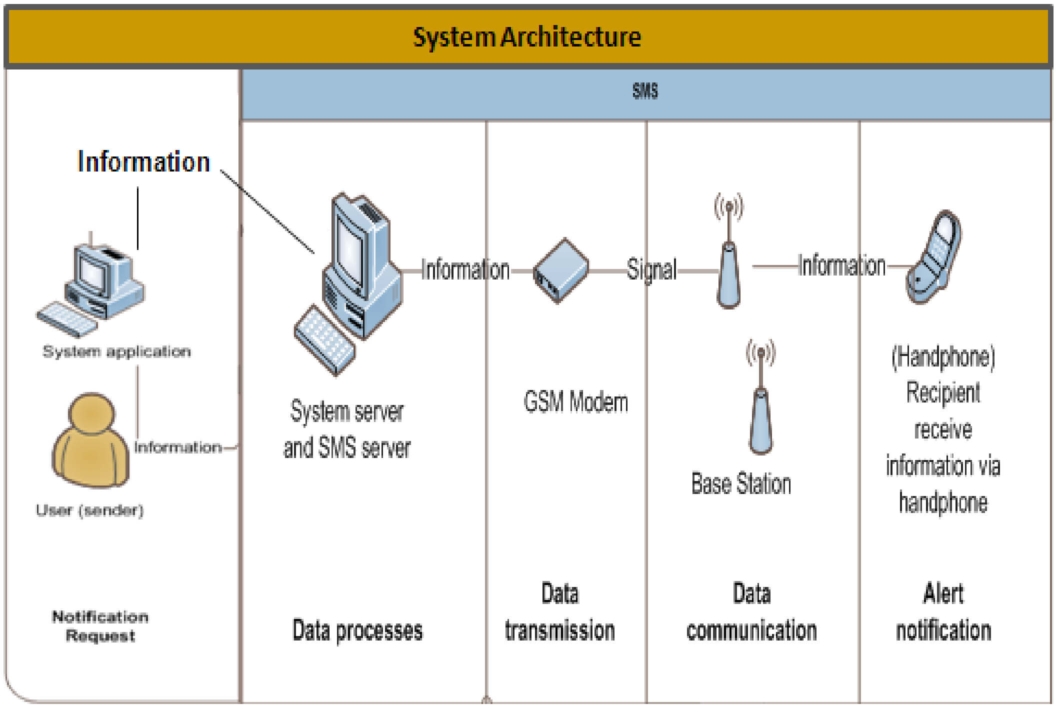


Figure 15 : SMS System Architecture

Architecture is related to how the system is going to interact with other components. Figure 15 shows the system architecture for this project using SMS. User will interact with system interface as a medium for sending information to the respective receiver. From system server, information is transmitted to GSM modem. GSM modem will send data signal to base station. The base station will transmit the SMS to the receiver via hand phone.

4.3 SMS System Interface

4.3.1 Log In

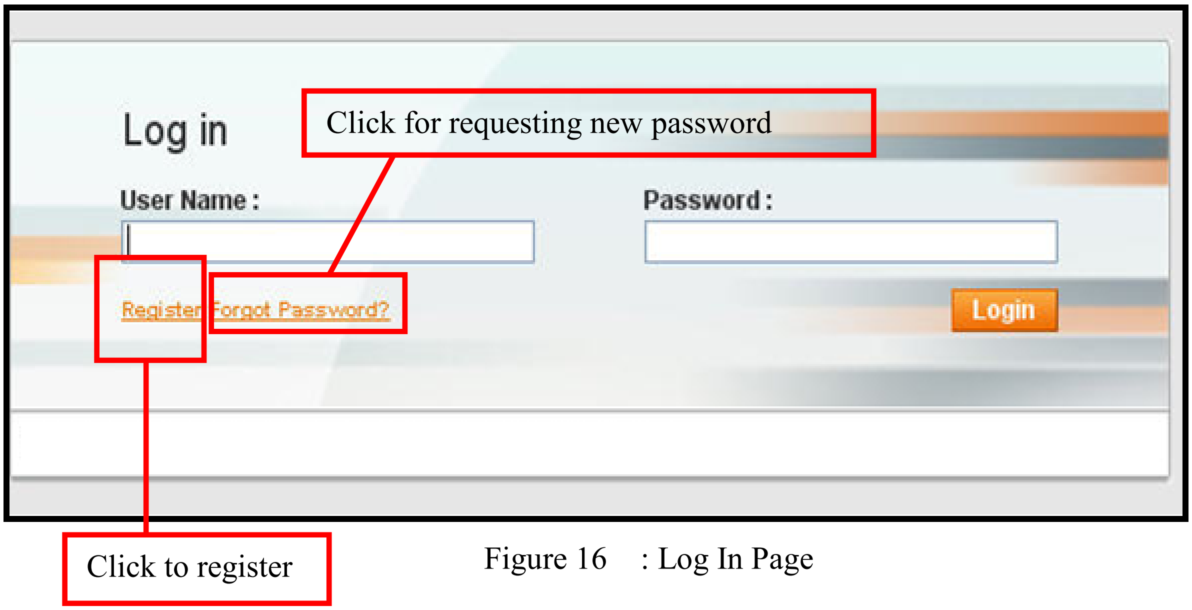


Figure 16 : Log In Page

User need to log in to the system in the beginning as shown in figure 16. Only user with valid username and password is able to use the system. New users need to register before enable to use the system. New user can click register button to proceed with system registration. If they forgot the password, just click forgot password button to request new password.

4.3.1 System Registration

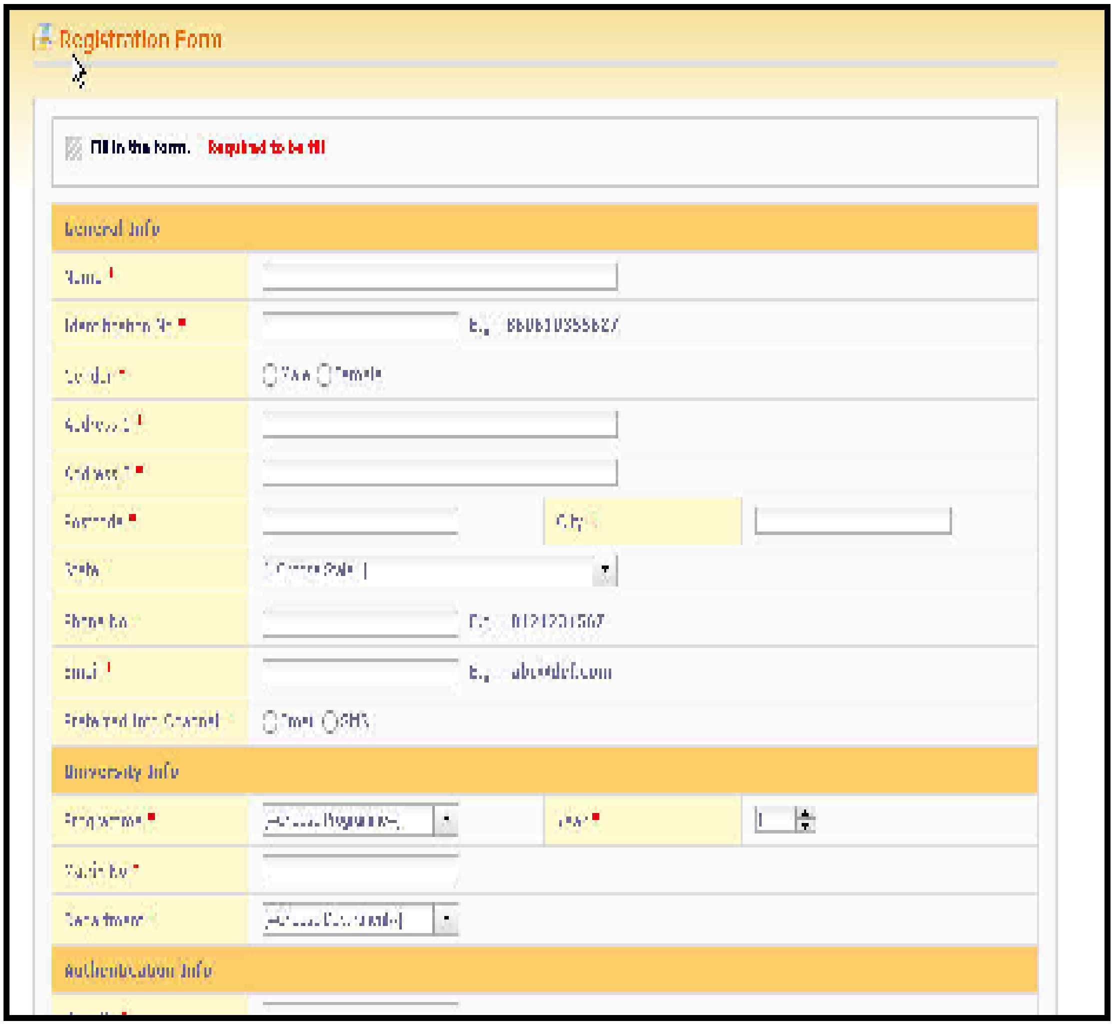


Figure 17 : Registration Form

New User need to fill in required information in the registration form as shown in figure 17. User need to click submit button after completeing the registration form. New user can be an active user after administrator have update his information status.

4.3.1 Main Page

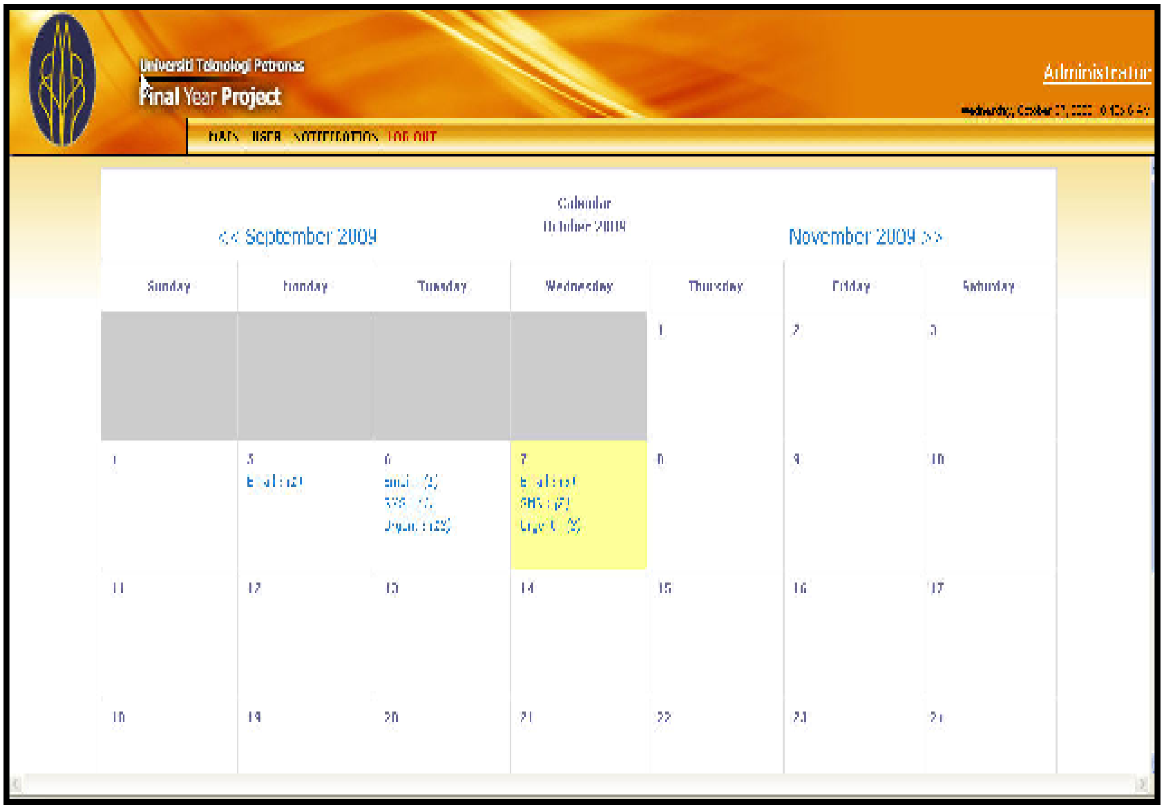


Figure 18 : Calendar

After successfully log in into the system, a calendar will be appeared as shown in figure 18. The calendar is located on the main page. The main purpose of putting the calendar in the main page is for user references in monitoring their daily test results. Each date column will appear the SMS notification.

4.3.1 End Receiver List

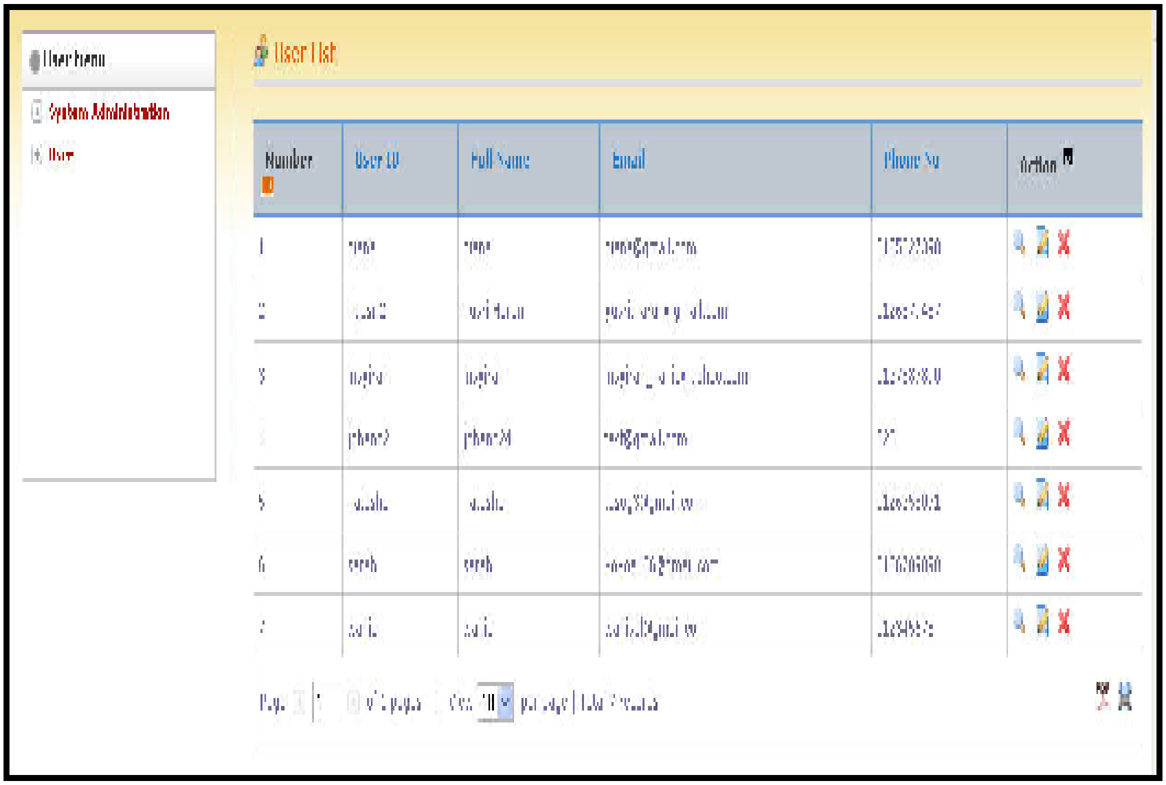
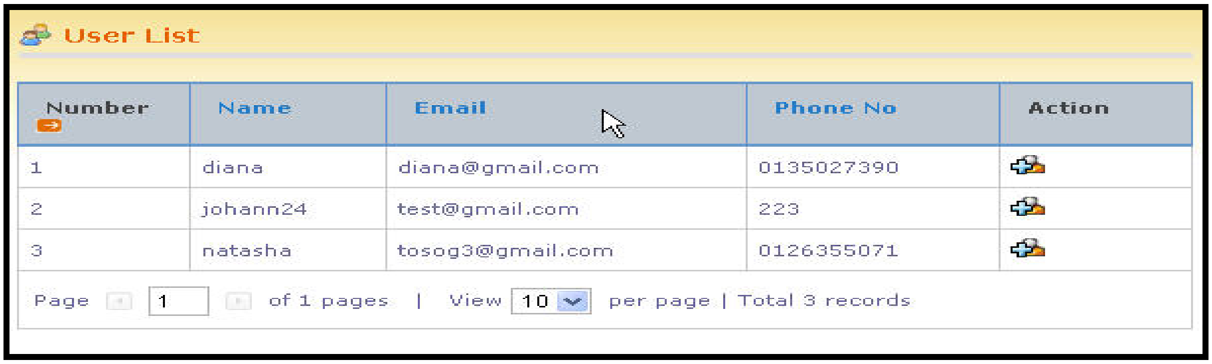


Figure 19 : End Receiver List

Under ‘USER’ tab, there are two types of user menu which are system administrator and user. System administrator is the list of system administrator details. System admin can be more than one person. System users list contain user details which are name and phone number as shown in figure 19. Administrator can click view, update or delete button on the right side of the list to update system user data.

4.3.1 Notification

User need to click the icon appears beside recipient column to select respective recipient. User needs to click send button to send SMS as shown in figure 20.



User List

Number	Name	Email	Phone No	Action
1	diana	diana@gmail.com	0135027390	
2	johann24	test@gmail.com	223	
3	natasha	tosog3@gmail.com	0126355071	

Page 1 of 1 pages | View 10 per page | Total 3 records

Figure 20 : Page for selecting SMS recipient

Figure 21 shows a notice indicating that the SMS is successfully sent to recipient will appear and the end receiver user will receive the SMS.

Figure 21 : Successfully send SMS notification

4.3.2 Log Out

User cans logout from the system by clicking logout label on the top of the page as shown figure 22. The system will back to the log in page after user log out.

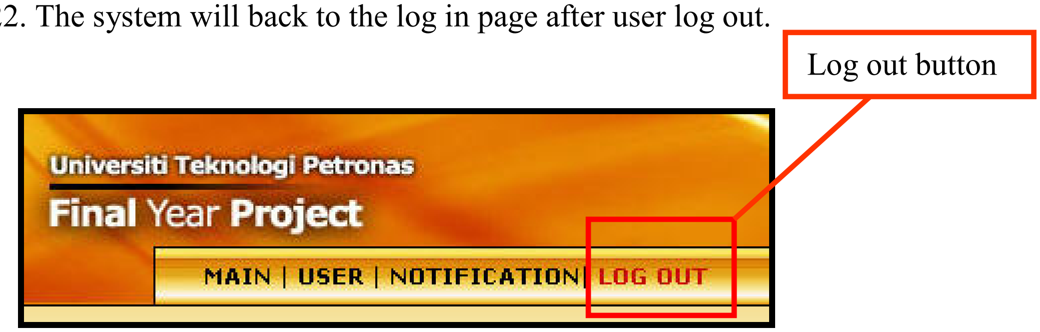


Figure 22 : Log Out

CHAPTER 5
CONCLUSION AND RECOMMENDATION

5.1 Conclusion

This development of the blood pressure monitoring device in a way that the result of each test can be sent to other people via SMS is essential in daily life and useful especially to those with the high blood pressure problem.

Apart from that, the development of this device is in the hope of becoming an important mobile device especially to those having blood pressure health problem and thus can help in monitoring their daily health.

5.2 Recommendation

It is recommended that in future that there will be more development that can be done to this device where there can be database application where the data obtained can be stored in a database automatically for future reference. The system can also be improved further by incorporating additional on-line capabilities to record data. This way the pressure and heart rate trend can be seen thus health monitoring would be easier.

REFERENCES

[1] http://www.drbloodpressure.com/05-mesurer8.html

[2] http://en.wikipedia.org/wiki/GSM_Modem

[3] http://www.Home blood pressure monitoring\Get the most out of home blood pressure monitoring.html

[4] Gary, V. K. and Wilkes, J.E., *Principles & Applications of GSM*, Prentice Hall, Upper Saddle River, NJ, 1999, pp. 20-21.

[5] Stefano Ceri, Florian Daniel, Maristella Matera, Federico M. Facca, “Model-driven Development of Context-Aware Web Application,” (1–32)

[6] “VB.net vs Java Comparison,”14 February 2009, http://theopensourcery.com/ vbjava2.htm

[7] Michael Calore, ”A History of Microsoft Windows,” October 2008 http://www.wired.com/gadgets/pcs/multimedia/2007/01/wiredphotos31

[8] Kenneth van Wyk, “Mozilla Firefox vs. Internet Explorer: Which is Safer?” September 2007

[9] Nicholas Economides and Evangelos Katsamakas, “Linux vs. Windows: A Comparison of Application and Platform innovation Incentives for Open Source and Proprietary Software Platform,” 2006.

[10] Jayson Pablo., “SMS vs Email: The Moot Point,” < http://www.buzzle.com/articles/sms-vs-email-the-moot-point.html > [Accessed on 12032007]

[11] Dr Daciana Illiescu and Dr Evor Hines. “SMS Based Student Feedback and Assessment,” SSchool of Engineering, University of Warwick. , 2006.

[12] TV Ramachandran.,” Evolution of GSM in India,” September 2005.

[13] Silicon Press, Third generation (3G) Wireless technology Brief,< http://www.silicon-press.com/briefs/brief.3g/index.html>

[14] http://www.freepatentsonline.com/6719703.html

[15] http://www.OscillometricMethod\Method for the measurement of the blood pressure.html

[16] http://www.severehypertension.net/hbp/more/oscillometric-method/

[17] http://www.blood-pressure-hypertension.com/how-to-measure/measure-blood-pressure-8.shtml

[18] http://www.Homebloodpressuremonitoring\Get the most out of home blood pressure monitoring.html

[19] Gary, V.K. and Wilkes, J.E., *Principles & Applications of GSM,* Prentice Hall, Upper Saddle River, NJ, 1999, pp. 20-21

[20] http://www.elemar.pl/mpx2052.pdf

APPENDICES

APPENDIX A

FYP I PROJECT MILESTONE

No.	Detail/ Week	1	2	3	4	5	6	7	8	9		10	11	12	13	14
1	Selection of Project Topic										Mid- Semester Break					
2	Preliminary Research Work															
3	Submission of Preliminary Report				●											
4	Seminar 1 (optional)															
5	Project Work															
6	Submission of Progress Report								●							
7	Seminar 2 (compulsory)															
8	Project work continues															
9	Submission of Interim Report Final Draft														●	
10	Oral Presentation															●

Indication:

●	Suggested milestone
	Process

APPENDIX B

FYP II PROJECT MILESTONE

No.	Detail/ Week	1	2	3	4	5	6	7	8	9		10	11	12	13	14
1	Project Work Continue										Mid- Semester Break					
2	Submission of Progress Report								●							
3	Seminar															
4	Project work continue															
5	Poster Exhibition											●				
6	Submission of Dissertation (soft bound)															●
7	Oral Presentation															●
8	Submission of Project Dissertation (Hard Bound)															●

Indication:

●	Suggested milestone
	Process

APPENDIX C

C++ CODING

```
#include <Mega32.h>
#include <delay.h>

#asm
    .equ __lcd_port = 0x15
#endasm

#include <math.h>
#include <lcd.h>
#include <stdio.h>
#include <stdlib.h>

//define states for motor control
#define startState 0
#define inflate1State 1
#define inflate2State 2
#define deflateState 3
#define displayState 4
#define resetState 5


//define states for Measure control
#define Sys_Measure 6
#define Sys_Cal 7
#define Rate_Measure 8
#define dias_Measure 9
#define dias_Cal 10

#define LCDwidth 16

void initialize(void);

//declare functions for motor control
void start_state(void);
void inflate1_state(void);
void inflate2_state(void);
void deflatestate(void);
void display_state(void);
void reset_state(void);
```

```
//declare all functions for measuring control
void pressuremeasure(void);
void sysmeasure(void);
void syscal(void);
void ratemeasure(void);
void diasmeasure(void);
void diascal(void);

//declare variable for motor controls
unsigned char Maybe0,Maybe1,Maybe2,countlcd;
unsigned char currentState;
unsigned int timepress0, timepress1, timepress2, timelcd;
char lcd_output[17];

//declare variable for measuring and calculating value
float DC_gain;
unsigned char meas_state;
unsigned int timing, timerate, timerun_dias, timecount, timedeflate, timedisplay;
float  maxpressure, pressure,accum_data, press_data;
unsigned char count, stop_count;

//ADC data variabls
float Vref;
unsigned char data;
float adc_data, former;

//define counter
unsigned char sys_count,count_average, countpulse;

//declare rate measure variable
float time_pulse,pulse_period, total_pulse_period, pulse_per_min;

//declare systolic and diastolic variable
float systolic, diastolic;

//declare all the threshold values
float TH_sys, TH_rate, TH_dias;
```

```
//*********************************************
//timer 0 compare ISR
interrupt [TIM0_COMP] void timer0_compare(void)
{
  if(~PINB & 0x01) timepress0++;
  if(~PINB & 0x02) timepress1++;
  if(~PINB & 0x04) timepress2++;
  timecount++;
  timedeflate++;
   //Decrement each time tast if they are not already zero

   //timing for sampling data at every 40 msec
         if(timing>0) --timing;
         //---------------------------------------------------
          //run time for different tasks

         //run timerate for measuring heart rate
          if(timerate<6000) ++timerate;

          //run timerun_dias
          if(timerun_dias<2000) ++timerun_dias;

  //if(countlcd) timelcd++;

   //run time for the display
   if(timedisplay<2000) ++timedisplay;
}

/*********************************************
// ADC Interrupt
//*******************************************************

interrupt [ADC_INT] void adc_complete(void)
{
 data = ADCH;
 //then calculate adc_data into float;
   adc_data = (float)(((float)data)/256*Vref);
```

```
//if signal is above threshold, go to calculate systolic pressure
 if(meas_state ==Sys_Measure)
          {
            if(former<=TH_sys && adc_data>TH_sys)
             sys_count++;

            former = adc_data;
   }
 //------------------------------------------------------------
 else if(meas_state==Sys_Cal)
          {

           if(count<4)
            {
            accum_data=accum_data+adc_data;
            count++;
            }
           if(count==4)
            {
           press_data=accum_data/4;
           systolic = (press_data/DC_gain)*9375;//calculate from adc_data
           meas_state = Rate_Measure;
           countpulse=0;
           former = 2.4; //set the initial point for rate measuring
           count_average=0;
            }
          }
 //------------------------------------------------------------
else if(meas_state==Rate_Measure)
 {
          if(count_average<5)
          {

           if(former<TH_rate && adc_data>TH_rate && countpulse==0)
                      {
             timerate=0;
             countpulse=1;
             former=adc_data;
             }
```

```
if(former<TH_rate && adc_data>TH_rate && countpulse==1)
            {
            total_pulse_period=total_pulse_period+timerate;
            timerate=0;
            count_average++; //finish reading one period
            }

          }//count_average

          former=adc_data;

}// else if(meas_state=Rate_Measure)
//------------------------------------------------------------
else if(meas_state==dias_Measure)
{
         if(timerun_dias<2000)
                 {
         if(adc_data>TH_dias)
         { timerun_dias=0; //reset time if the signal
         //is still greater than threshold (so it will never reach 1999)
         //if it doesn't reset,the time will stuck at 1999
         lcd_clear();
                 lcd_gotoxy(0,0);
                 lcd_putsf("Dias measure");
                 }
                 }
         if(timerun_dias>=2000)
                 {
                 meas_state = dias_Cal;//if done go back to Sys_Measure to be ready for
next opt
                 }

}
//------------------------------------------------------------
else if(meas_state==dias_Cal)
{
    diastolic = (adc_data/DC_gain)*9375;//calculate from adc_data
          meas_state = Sys_Measure;
          currentState = displayState;
          //open valve
          PORTD=0;

}
```

```
timing = 40;//set time for another conversion

}// end of ADC interrupt
//**********************************************************
void main(void)
{
        initialize();
        while(1)
        {
                switch(currentState)
                {
                        case startState:
                                start_state();
                                break;
                        case inflate1State:
                                inflate1_state();
                                break;
                        case inflate2State:
                                inflate2_state();
                                break;
                        case deflateState:
                                deflatestate();
                                break;
                        case displayState:
                                display_state();
                                break;
                        case resetState:
                                reset_state();
                                break;
                }
        }
}
//**********************************************
void start_state(void)
{   sys_count=0;
   pressure = 0;
   accum_data=0;
        press_data=0;
   count=0;
   stop_count=0;
maxpressure = 160;
   meas_state = Sys_Measure;
   former=TH_sys-0.01;
```

```
timerun_dias=0;
        time_pulse=0;
        timerate=0;

        timing=40;

        total_pulse_period=0;
        systolic=0;
        diastolic=0;
        pulse_per_min=0;

        sys_count=0;
        count_average=0;
        countpulse=0;

        if((~PINB & 0x01) && (timepress0 > 30)) Maybe0 = 1;
        if(Maybe0 && (PINB == 0xff))

        {
                        countlcd = 1;
                        timelcd = 0;
                        lcd_clear();
                        lcd_gotoxy(0,0);
                        lcd_putsf("Inflating");
                        currentState = inflate1State;
                        Maybe0 = 0;
                        timepress0 = 0;
                        timecount=0;
                        //turn on motor and close the valve
                        PORTD=0x03;
                        //activate ADC
        }
}
//*********************************************

void inflate1_state(void)
{
        if(timecount>=200)
        {
        timecount=0;
        sprintf(lcd_output,"%-i",(int)pressure);
        lcd_gotoxy(0,1);
        lcd_puts(lcd_output);
        }
```

```
{
                lcd_clear();
                lcd_gotoxy(0,0);
                lcd_putsf("Emergency Stop");
                sprintf(lcd_output,"%-i",(int)pressure);
            lcd_gotoxy(0,1);
            lcd_puts(lcd_output);
                //turn off motor and open the valve
                PORTD=0;
                currentState = resetState;
                Maybe1 = 0;
                timepress1 = 0;
                countlcd = 0;
        }
        else
        {
                currentState = inflate2State;
        }

}
//*********************************************
void inflate2_state(void)
{
    ADMUX=0b00100001;//choose ADC1 for reading DC

   //enable ADC and set prescaler to 1/128*16MHz=125,000
   //and uncheck interupt enable
   //and start a conversion
   ADCSR = 0b11000111;
   data= ADCH;
   adc_data = (float)(((float)data)/256*Vref);
   pressure= (adc_data/DC_gain)*9375;

   if(pressure>=maxpressure) stop_count++;
   else stop_count = 0;

   if(stop_count>=5)
{
                lcd_clear();
                lcd_gotoxy(0,0);
                lcd_putsf("Deflating");
                sprintf(lcd_output,"%-i",(int)pressure);
            lcd_gotoxy(0,1);
```

```
lcd_puts(lcd_output);
                //turn off motor but keep the valve
                PORTD = 0x02;
                delay_ms(1000);
                currentState = deflateState;
                timedeflate = 0;
                sprintf(lcd_output,"%-i",(int)pressure);
           lcd_gotoxy(0,1);
           lcd_puts(lcd_output);
        }
        else
        {
                currentState = inflate1State;
        }

}
//*********************************************
void deflatestate(void)
{
  /*if(timedeflate >= 1900)
  {
     PORTD = 0;
     if(timedeflate >= 2000)
     {
       PORTD = 0x02;
       timedeflate = 0;
     }
  }*/
        if((~PINB & 0x02) && (timepress1 > 30)) Maybe1 = 1;
        if(Maybe1 && (PINB == 0xff))
        {
        lcd_clear();
                lcd_gotoxy(0,0);
                lcd_putsf("Emergency Stop");
                sprintf(lcd_output,"%-i",(int)pressure);
           lcd_gotoxy(0,1);
           lcd_puts(lcd_output);
                //turn off motor and open the valve
                PORTD=0;
                currentState = resetState;
                Maybe1 = 0;
                timepress1 = 0;
  }
```

```
//if(done) --> Display state
          if(currentState==deflateState) pressuremeasure(); //if still deflating, measure
everything
}

//********************************************
void display_state(void)
{
if(timedisplay<=1000)
{
   if(timecount>=200)
          {
          lcd_clear();
   timecount=0;
          lcd_gotoxy(0,0);
          lcd_putsf("Sys");
          lcd_gotoxy(7,0);
          lcd_putsf("Dias");
          lcd_gotoxy(15,0);
          lcd_putsf("HR");
          sprintf(lcd_output,"%-i",(int)systolic);
          lcd_gotoxy(0,1);
          lcd_puts(lcd_output);

          sprintf(lcd_output,"%-i",(int)diastolic);
          lcd_gotoxy(7,1);
          lcd_puts(lcd_output);

          sprintf(lcd_output,"%-i",(int)pulse_per_min);
          lcd_gotoxy(14,1);
          lcd_puts(lcd_output);
          }
}
else if (timedisplay>1000&&timedisplay<2000)
{
          if(timecount>=200)
          {         lcd_clear();
      timecount=0;

                    lcd_gotoxy(0,0);
                    lcd_putsf("Black: Resume");

          }
}
```

```
else
{
   timedisplay=0;
}
         if((~PINB & 0x04) && (timepress2 > 30)) Maybe2 = 1;
         if(Maybe2 && (PINB == 0xff))
         {
                  lcd_clear();
                  lcd_gotoxy(0,0);
                  lcd_putsf("White: Start");
                  lcd_gotoxy(0,1);
                  lcd_putsf("Grey: Stop");
                  currentState = startState;
                  timepress2 = 0;
                  Maybe2=0;
                  systolic=0;
                  diastolic=0;
                  pulse_per_min=0;

         }

}
//*********************************************
void reset_state(void)
{
if(timedisplay<=1000)
{
   if(timecount>=200)
         {   timecount=0;
         lcd_clear();
                  lcd_gotoxy(0,0);
                  lcd_putsf("Emergency Stop");
         }
}
else if (timedisplay>1000&&timedisplay<2000)
{
         if(timecount>=200)
         {        lcd_clear();
      timecount=0;

                  lcd_gotoxy(0,0);
                  lcd_putsf("Black: Resume");

         }
}
```

```
else
{
   timedisplay=0;
}
         if((~PINB & 0x04) && (timepress2 > 30)) Maybe2 = 1;
         if(Maybe2 && (PINB == 0xff))
         {
                  lcd_clear();
                  lcd_gotoxy(0,0);
                  lcd_putsf("White: Start");
                  lcd_gotoxy(0,1);
                  lcd_putsf("Grey: Stop");
                  currentState = startState;
                  timepress2 = 0;
                  Maybe2=0;
         }

}
//**********************************************
// Function to measure everything
//----------------------------------------------
void pressuremeasure(void)
{
 switch (meas_state)
         {
            case Sys_Measure:
                     if(timing==0) sysmeasure(); //sampling signal at 40msec
                     break;

            case Sys_Cal:
                 if(timing==0) syscal();
                 break;

            case Rate_Measure:
                     if(timing==0) ratemeasure();
                     break;

            case dias_Measure:
                     diasmeasure();
                     break;

            case dias_Cal:
                     diascal();
                     break;
```

```
 } //switch

}//pressuremeasure
//*********************************************************
void sysmeasure(void)
{
        if(timing==0)
                {ADMUX = 0b00100000; //choose ADC0 for reading AC

      //enable ADC and set prescaler to 1/128*16MHz=125,000
      //and set interupt enable
      //and start a conversion
      ADCSR = 0b11001111;

      }
       if(sys_count>=6)
       {
       meas_state = Sys_Cal;
       timecount=0;
       }

      if(timecount>=200)
      {
      lcd_clear();
                lcd_gotoxy(0,0);
                lcd_putsf("Measuring");
      timecount=0;
      }
}

//**********************************************************
//this function is to calculate systolic pressure
void syscal(void)
{

                ADMUX=0b00100001;//choose ADC1 for reading DC

                //enable ADC and set prescaler to 1/128*16MHz=125,000
      //and set interupt enable
      //and start a conversion
      ADCSR = 0b11001111;

      }
```

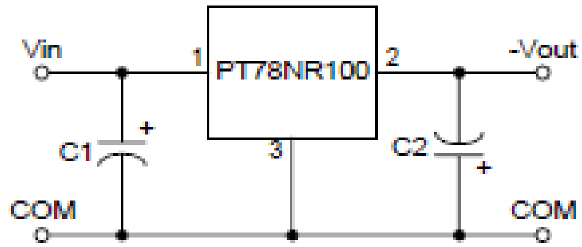
```
    if(timecount>=200)
      {
      lcd_clear();
                lcd_gotoxy(0,0);
                lcd_putsf("Sys Cal");
      timecount=0;
      }


}//syscal

//***********************************************************
void ratemeasure(void)
{

                ADMUX=0b00100000; //choose ADC0 for reading AC

                //enable ADC and set prescaler to 1/128*16MHz=125,000
      //and set interupt enable
      //and start a conversion
      ADCSR = 0b11001111;
      //calculate the mean of pulse rate
           if(count_average==5)
           {
            pulse_period = total_pulse_period/5000;
            pulse_per_min= 60/pulse_period;

           lcd_clear();
                lcd_gotoxy(0,0);
                lcd_putsf("Pulse Rate");
           sprintf(lcd_output,"%-i",(int)pulse_per_min);
      lcd_gotoxy(0,1);
      lcd_puts(lcd_output);

            meas_state = dias_Measure;
            //then set timerun_dias=0
            //also reset count_average for the next operation
            count_average=0;
            timerun_dias=0;
           }
}
//***********************************************************
```

```
   void diasmeasure(void)
{
                 ADMUX=0b00100000;//choose ADC1 for reading AC

                 //enable ADC and set prescaler to 1/128*16MHz=125,000
      //and set interupt enable
      //and start a conversion
      ADCSR = 0b11001111;
}//dias measure
//**********************************************************
void diascal(void)
{
                 ADMUX=0b00100001;//choose ADC1 for reading DC

                 //enable ADC and set prescaler to 1/128*16MHz=125,000
      //and set interupt enable
      //and start a conversion
      ADCSR = 0b11001111;

      if(timecount>=200)
      {
      lcd_clear();
                 lcd_gotoxy(0,0);
                 lcd_putsf("Dias_Cal");
      timecount=0;
      }
}
void initialize(void)
{


         //Initialize LCD
         timecount=0;
         lcd_init(LCDwidth);
         lcd_clear();
         lcd_gotoxy(0,0);
         lcd_putsf("White: Start");
         lcd_gotoxy(0,1);
         lcd_putsf("Grey: Stop");
//set up timer0
   TIMSK =2; //turn on timer 0 comp match
   OCR0 = 250; //set the compare register to 250
//prescaler to 64 and turn on clear-on-match
   TCCR0 = 0b00001011;
   timepress0 = 0;
   timepress1 = 0;
```

```
  DDRB=0x00; //PORT B is an input(2 buttons)
   DDRD=0xff; //PORT D is an output(motor control);
   PORTD=0x00;
   PORTB=0xff;
   PORTA=0x00;

   maxpressure = 160;
   meas_state = Sys_Measure;
   former=TH_sys-0.01;


         TH_sys=4.0;
         TH_rate = 2.5;
         TH_dias = 4.8;
         timerun_dias=0;
         time_pulse=0;
         timerate=0;
         timing=40;
         timedisplay=0;

         total_pulse_period=0;
         systolic=0;
         diastolic=0;
         pulse_per_min=0;
         Vref=5.0;


         sys_count=0;
         count_average=0;
         countpulse=0;

         DC_gain=213;

         accum_data=0;
         press_data=0;
         count=0;


         #asm
         sei
         #endasm
}
```

APPENDIX D

VOLTAGE REGULATOR SCHEMATICS AND SPECIFICATION



C1 = Required 100µF electrolytic (1)
C2 = Required 100µF electrolytic (1)

Figure D1: The Voltage Regulator Schematic

Characteristics (T_a= 25°C unless noted)	Symbols	Conditions	PT78NR100 SERIES Min	Typ	Max	Units
Output Current	I_o	Over V_{in} range V_o=-5V V_o=-6V V_o=-7, -8, -9V V_o=-10V V_o=-12V V_o=-13.9, -15V	0.05 (2) 0.05 (2) 0.05 (2) 0.05 (2) 0.05 (2) 0.05 (2)	— — — — — —	1.00 0.8 0.55 0.5 0.40 0.30	A
Short Circuit Current	I_{sc}	V_{in}=10V	—	$4\times I_{max}$	—	Apk
Inrush Current	I_{ir} t_{ir}	V_{in}=10V On start-up	— —	4 0.5	— —	A mSec
Input Voltage Range	V_{in}	$0.1 \le I_o \le I_{max}$ V_o=-5V V_o=-6, -7, -8, -9V V_o=-10, -12V V_o=-13.9, -15V	7 7 7 7	— — — —	25 21 18 15	V V V V
Output Voltage Tolerance	ΔV_o	Over V_{in} range T_a=-20°C to +70°C	—	±1.0	±3.0	%V_o
Line Regulation	Reg_{line}	Over V_{in} range	—	±0.5	±1.0	%V_o
Load Regulation	Reg_{load}	$0.1 \le I_o \le I_{max}$	—	±0.5	±1.0	%V_o
V_o Ripple/Noise	V_n	V_{in}=10V, I_o=I_{max}	—	±2	—	%V_o
Transient Response (with 100µF output cap)	t_{tr}	50% load change V_o over/undershoot	— —	100 5.0	250 —	µSec %V_o
Efficiency	η	V_{in}=10V, I_o=$0.5\times I_{max}$, V_o = -5V	—	75	—	%
Switching Frequency	f_o	Over V_{in} and I_o ranges	600	650	700	kHz
Absolute Maximum Operating Temperaturte Range	T_a	Free Air Convection, (40-60LFM) Over V_{in} Range	-40	—	+85 (3)	°C
Thermal Resistance	θ_{ja}	Free Air Convection, (40-60LFM)	—	45	—	°C/W
Storage Temperature	T_s	—	-40	—	+125	°C
Mechanical Shock	—	Per Mil-STD-883D, Method 2002.3	—	500	—	G's
Mechanical Vibration	—	Per Mil-STD-883D, Method 2007.2, 20-2000 Hz, soldered in a PC board	—	5	—	G's
Weight	—	—	—	6.5		Grams

Table D1: The Voltage Regulator Specification

APPENDIX E

AMPLIFIER SCHEMATICS AND SPECIFICATION

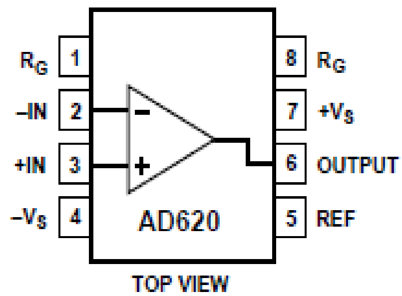


Figure E1: The Amplifier Schematic

SPECIFICATIONS

Typical @ 25°C, V_S = ±15 V, and R_L = 2 kΩ, unless otherwise noted.

Table 1.

Parameter	Conditions	AD620A			AD620B			AD620S[1]			Unit
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
GAIN	$G = 1 + (49.4\ k\Omega/R_G)$										
Gain Range		1		10,000	1		10,000	1		10,000	
Gain Error[2]	V_{OUT} = ±10 V										
G = 1			0.03	0.10		0.01	0.02		0.03	0.10	%
G = 10			0.15	0.30		0.10	0.15		0.15	0.30	%
G = 100			0.15	0.30		0.10	0.15		0.15	0.30	%
G = 1000			0.40	0.70		0.35	0.50		0.40	0.70	%
Nonlinearity	V_{OUT} = –10 V to +10 V										
G = 1–1000	R_L = 10 kΩ		10	40		10	40		10	40	ppm
G = 1–100	R_L = 2 kΩ		10	95		10	95		10	95	ppm
Gain vs. Temperature											
	G = 1			10			10			10	ppm/°C
	Gain >1[2]			–50			–50			–50	ppm/°C
VOLTAGE OFFSET	(Total RTI Error = $V_{OSI} + V_{OSO}/G$)										
Input Offset, V_{OSI}	V_S = ±5 V to ± 15 V		30	125		15	50		30	125	µV
Overtemperature	V_S = ±5 V to ± 15 V			185			85			225	µV
Average TC	V_S = ±5 V to ± 15 V		0.3	1.0		0.1	0.6		0.3	1.0	µV/°C
Output Offset, V_{OSO}	V_S = ±15 V		400	1000		200	500		400	1000	µV
	V_S = ± 5 V			1500			750			1500	µV
Overtemperature	V_S = ±5 V to ± 15 V			2000			1000			2000	µV
Average TC	V_S = ±5 V to ± 15 V		5.0	15		2.5	7.0		5.0	15	µV/°C
Offset Referred to the Input vs. Supply (PSR)	V_S = ±2.3 V to ±18 V										
G = 1		80	100		80	100		80	100		dB
G = 10		95	120		100	120		95	120		dB
G = 100		110	140		120	140		110	140		dB
G = 1000		110	140		120	140		110	140		dB
INPUT CURRENT											
Input Bias Current			0.5	2.0		0.5	1.0		0.5	2	nA
Overtemperature				2.5			1.5			4	nA
Average TC			3.0			3.0			8.0		pA/°C
Input Offset Current			0.3	1.0		0.3	0.5		0.3	1.0	nA
Overtemperature				1.5			0.75			2.0	nA
Average TC			1.5			1.5			8.0		pA/°C
INPUT											
Input Impedance											
Differential			10‖2			10‖2			10‖2		GΩ_pF
Common-Mode			10‖2			10‖2			10‖2		GΩ_pF
Input Voltage Range[3]	V_S = ±2.3 V to ±5 V	$–V_S + 1.9$		$+V_S – 1.2$	$–V_S + 1.9$		$+V_S – 1.2$	$–V_S + 1.9$		$+V_S – 1.2$	V
Overtemperature		$–V_S + 2.1$		$+V_S – 1.3$	$–V_S + 2.1$		$+V_S – 1.3$	$–V_S + 2.1$		$+V_S – 1.3$	V
	V_S = ± 5 V to ±18 V	$–V_S + 1.9$		$+V_S – 1.4$	$–V_S + 1.9$		$+V_S – 1.4$	$–V_S + 1.9$		$+V_S – 1.4$	V
Overtemperature		$–V_S + 2.1$		$+V_S – 1.4$	$–V_S + 2.1$		$+V_S + 2.1$	$–V_S + 2.3$		$+V_S – 1.4$	V

Table E1: The Amplifier Specification

APPENDIX F

PRESSURE INTERFACE HARWARE LIST AND SCHEMATIC

Table F1: The Pressure Interface Hardware List

No.	Component	Product Code	Unit
1	IC PIC16F876A	IC-PIC-16F876A	1
2	IC Socket-28 pin(silm)	IS-28PIN(S)	1
3	Crystal H49S (Low Profile) 20MHz	CR-H49S-20M	1
4	Voltage Regulator +5V	VR-7805	1
5	Diode 1N4007	DI-1N4007	1
6	Diode 1N4148	DI-1N4148	1
7	Electrolytic Capacitor 16V 100uF	CP-EC-16-100	3
8	Ceramic Capacitor 0.1uF	CP-CC-0.1UF	4
9	Ceramic Capacitor 30pF	CP-CC-30PF	2
10	2510 PCB Connector 2 Ways (R/A)	CN-02-2510RA	1
11	Slide Switch 3 Pins Black	SW-SL-3N-061206	1
12	DC Plug (Adaptor Socket)	CN-HL-2527-B	1
13	6x6x1 Push Button 4Pin	SW-PBM-4N-060601	4
14	Zener Diodes 1/2W 3V3	DI-ZD-05W-3V3	3
15	2 x 16 character LCD	DS-LCD-JHD162A	1
16	LED 3mm Green	DS-LED-3NG	1
17	Resistor 1/4W 330R	RS-025W-330R	1
18	Resistor 1/4W 10K	RS-025W-10K	4
19	Resistor 1/4W 1K	RS-025W-1K	9
20	Resistor 1/4W 4K7	RS-025W-4K7	2
21	Straight Pin Header(Male) 16 Ways	CN-PH-M140S	1
22	Straight Pin Header(Male) 7 Ways	CN-PH-M140S	1
23	Straight Pin Header(Male) 4 Ways	CN-PH-M140S	2
24	Transistor 2N2222	TR-2N-2222	2
25	Voltage Regulator +3.3V	VR-1117-3.3	1
26	10 Ways Straight Box Header	CN-IDC-BOX-10	1
27	PCB for PR22	PCB-PR-022	1
	Optional		
28	AC to DC adaptor	AC to DC adaptor	1
29	UIC00A Programmer	UIC00A Programmer	1
30	AD7746 Capacitive Sensor	SN-AD7746	1
31	HMC6352 Compass Module	SN-HMC6352	1
32	SCP1000-D01 Barometric Sensor	SN-SCP1000	1

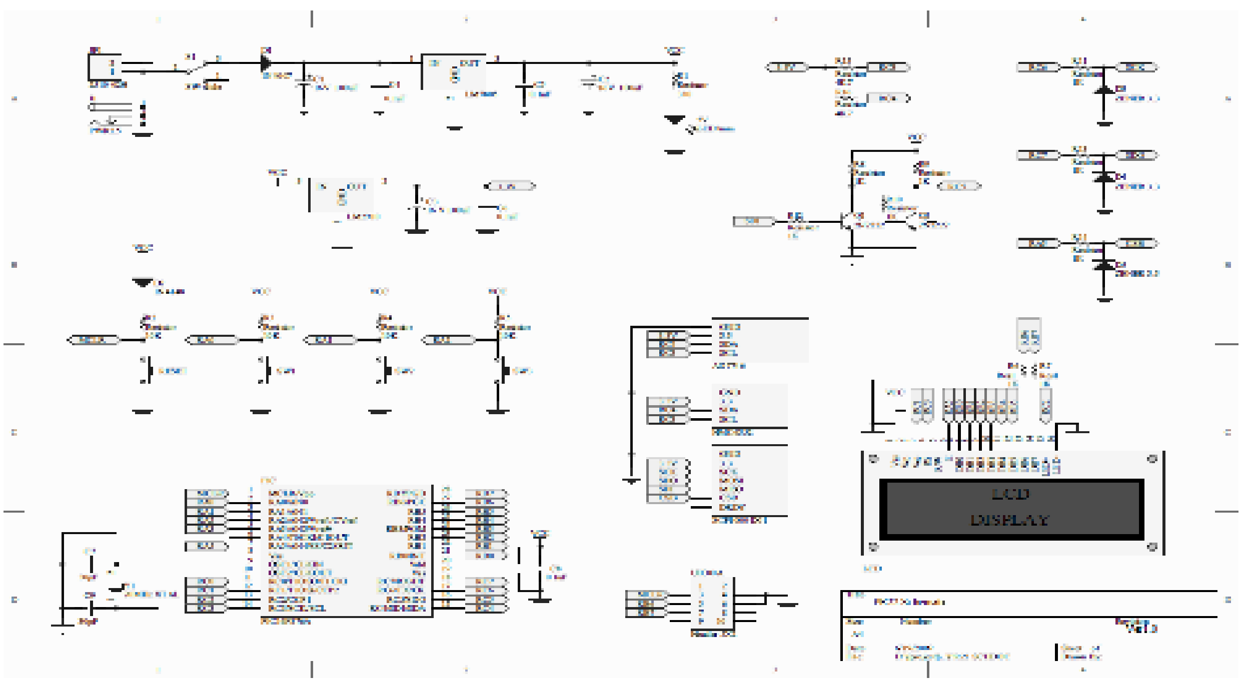
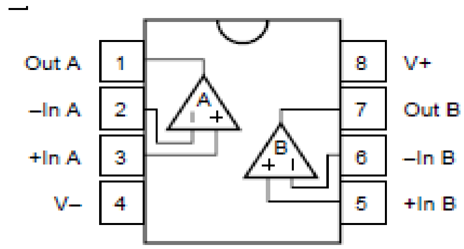


Figure F1: The Interface Schematic

APPENDIX G

OPERATIONAL AMPLIFIER SCHEMATICS AND SPECIFICATION



8-Pin DIP, SO-8

Figure G1: The Operational Amplifier Schematic

SPECIFICATIONS: V_S = ±5V to V_S = ±15V

At T_A = +25°C, and R_L = 2kΩ, unless otherwise noted.
Boldface limits apply over the specified temperature range, –40°C to +85°C.

PARAMETER	CONDITION	OPA277P, U OPA2277P, U			OPA277PA, UA OPA2277PA, UA OPA4277PA, UA			UNITS
		MIN	TYP(1)	MAX	MIN	TYP(1)	MAX	
OFFSET VOLTAGE								
Input Offset Voltage: V_{OS}								
OPA277P, U (high grade, single)			±10	±20				μV
OPA2277P, U (high grade, dual)			±10	±25				μV
All PA, UA Versions						±20	±50	μV
Input Offset Voltage Over Temperature								
OPA277P, U (high grade, single)	T_A = –40°C to +85°C			±30				μV
OPA2277P, U (high grade, dual)	T_A = –40°C to +85°C			±50				μV
All PA, UA Versions	T_A = –40°C to +85°C						±100	μV
Input Offset Voltage Drift dV_{OS}/dT								
OPA277P, U (high grade, single)	T_A = –40°C to +85°C		±0.1	±0.15				μV/°C
OPA2277P, U (high grade, dual)	T_A = –40°C to +85°C		±0.1	±0.25				μV/°C
All PA, UA Versions	T_A = –40°C to +85°C					±0.15	±1	μV/°C
Input Offset Voltage: (all models)								
vs Time			0.2			*		μV/mo
vs Power Supply PSRR	V_S = ±2V to ±18V		±0.3	±0.5		*	±1	μV/V
T_A = –40°C to +85°C	V_S = ±2V to ±18V			±0.5			±1	μV/V
Channel Separation (dual, quad)	dc		0.1			*		μV/V
INPUT BIAS CURRENT								
Input Bias Current I_B			±0.5	±1		*	±2.8	nA
T_A = –40°C to +85°C				±2			±4	nA
Input Offset Current I_{OS}			±0.5	±1		*	±2.8	nA
T_A = –40°C to +85°C				±2			±4	nA
NOISE								
Input Voltage Noise, f = 0.1 to 10Hz			0.22			*		μVp-p
			0.035			*		μVrms
Input Voltage Noise Density, f = 10Hz e_n			12			*		$nV/\sqrt{Hz}$
f = 100Hz			8			*		$nV/\sqrt{Hz}$
f = 1kHz			8			*		$nV/\sqrt{Hz}$
f = 10kHz			8			*		$nV/\sqrt{Hz}$
Current Noise Density, f = 1kHz i_n			0.2			*		$pA/\sqrt{Hz}$
INPUT VOLTAGE RANGE								
Common-Mode Voltage Range V_{CM}		(V–) +2		(V+) –2	*		*	V
Common-Mode Rejection CMRR	V_{CM} = (V–) +2V to (V+) –2V	130	140		115	*		dB
T_A = –40°C to +85°C	V_{CM} = (V–) +2V to (V+) –2V	128			115			dB
INPUT IMPEDANCE								
Differential			100 ‖ 3			*		MΩ ‖ pF
Common-Mode	V_{CM} = (V–) +2V to (V+) –2V		250 ‖ 3			*		GΩ ‖ pF
OPEN-LOOP GAIN								
Open-Loop Voltage Gain A_{OL}	V_O = (V–)+0.5V to (V+)–1.2V, R_L = 10kΩ		140			*		dB
	V_O = (V–)+0.5V to (V+)–1.5V, R_L = 2kΩ	126	134		*	*		dB
T_A = –40°C to +85°C	V_O = (V–)+0.5V to (V+)–1.5V, R_L = 2kΩ	126			*			dB
FREQUENCY RESPONSE								
Gain-Bandwidth Product GBW			1			*		MHz
Slew Rate SR			0.8			*		V/μs
Settling Time, 0.1%	V_S = ±15V, G = 1, 10V Step		14			*		μs
0.01%	V_S = ±15V, G = 1, 10V Step		16			*		μs
Overload Recovery Time	V_{IN} • G = V_S		3			*		μs
Total Harmonic Distortion + Noise THD+N	1kHz, G = 1, V_O = 3.5Vrms		0.002			*		%
OUTPUT								
Voltage Output V_O	R_L = 10kΩ	(V–) +0.5		(V+) –1.2	*		*	V
T_A = –40°C to +85°C	R_L = 10kΩ	(V–) +0.5		(V+) –1.2	*		*	V
	R_L = 2kΩ	(V–) +1.5		(V+) –1.5	*		*	V
T_A = –40°C to +85°C	R_L = 2kΩ	(V–) +1.5		(V+) –1.5	*		*	V
Short-Circuit Current I_{SC}			±35			*		mA
Capacitive Load Drive C_{LOAD}			See Typical Curve			*		

Table G1: The Operational Amplifier Specification